

REMARKS

The Applicant and the undersigned thank Examiner Guill for his careful review of this application. The Applicant especially thanks Examiner Guill and Supervisory Patent Examiner (SPE) Teska for their time and consideration given during the telephonic interview of May 2, 2005. A summary of this telephonic interview is provided below.

Claims 1-10 have been rejected by the Examiner. Upon entry of this amendment, Claims 11-30 remain pending in this application. The independent claims are Claims 11, 20, and 29.

Consideration of the present application is respectfully requested in light of the above claim amendments to the application, the telephonic interview, and in view of the following remarks.

Summary of Telephonic Interview of May 2, 2005

The Applicant and the undersigned thank the Examiners for their time and consideration given during the telephonic interview of May 2, 2005. During this telephonic interview, a proposed amendment to the claims was discussed. The Applicant provided the proposed amendment to the claims in advance of the interview.

The Applicant's representative explained to Examiner Guill that the prior art of record does not provide any teaching of automatically configuring a virtual machine to execute a target program in one of three modes of operation. It was explained that the prior art of record does not check to determine what type of target program is to be analyzed and based on that determination, automatically configuring the virtual machine to run the target program in one of three modes of operation: a first mode of operation comprising a real mode, a second mode of operation for executing a target program comprising a high level programming language, and a third mode of operation comprising a protected mode for executing a target program comprising thirty-two bit code.

Examiner Guill provided a few suggestions to the claims in order to make them more clear under 35 U.S.C. §112, second paragraph. The Applicant's representative agreed to those suggestions and they have been adopted in this paper.

U.S. Pat. No. 6,851,057 issued in the name of Nachenberg on February 1, 2005 (hereinafter, the "Nachenberg reference") was briefly discussed. Examiner Guill had informed the Applicant's representative of the Nachenberg reference prior to the telephonic interview.

The Applicant's representative explained that the Nachenberg reference describes a virus detection system that uses virus signatures and partial emulation of target code. It was explained that the Nachenberg reference does not use a virtual machine and therefore, this reference does not automatically configure a virtual machine to execute a target program in one of three modes of operation.

Examiner Guill expressed that he understood the Applicant's representative comments and the concepts presented by the new claims. Examiner Guill indicated that an update search would be conducted after the Applicant submits the claims in a formal amendment.

The Applicant and the undersigned request Examiner Guill to review this interview summary and to approve it by writing "Interview Record OK" along with his initials and the date next to this summary in the margin as discussed in MPEP § 713.04, p. 700-202.

Trademarks in the Specification

The Examiner noted that the Applicant has used some trademarks in the specification. The Applicant has capitalized these trademarks and provided generic terminology adjacent to these marks in accordance with the Examiner's helpful comments.

Claim Rejections under 35 U.S.C. §112, first paragraph

The Examiner rejected Claims 1-10 under 35 U.S.C. §112, first paragraph as failing to comply with the enablement requirement. The Examiner states that the claims contain subject matter which was not described in the specification in such a way as to enable one skilled in the art to make or use the invention. Specifically, the Examiner noted that the terms WINDOWS, DOS, LINUX, PENTIUM, VISUAL BASIC, and PERL are references to generic computer programs or hardware and they do not clearly identify the piece of software or hardware used.

The Applicant respectfully traverses these rejections. Because Claims 1-10 have been cancelled by the Applicant, the Applicant respectfully submits that the rejections of these claims under 35 U.S.C. § 112, first paragraph have been rendered moot. Accordingly, reconsideration and withdrawal of these rejections are respectfully requested.

Claim Rejections under 35 U.S.C. §101

The Examiner rejected Claims 1-10 under 35 U.S.C. §101 because the claimed invention is directed to non-statutory subject matter. The Examiner states that the preamble of a, "virtual machine system having a software processor..." is not a process, machine, manufacture, or composition of matter.

The Applicant respectfully traverses these rejections. Because Claims 1-10 have been cancelled by the Applicant, the Applicant respectfully submits that the rejections of these claims under 35 U.S.C. §101 have been rendered moot. Accordingly, reconsideration and withdrawal of these rejections are respectfully requested.

Claim Rejections Under 35 U.S.C. § 103

The Examiner rejected Claim 1 under 35 U.S.C. § 103(a) as being unpatentable over a 1997 article entitled, "A Generic Virus Detection Agent on the Internet," authored by Jieh-Sheng Lee et al. (hereinafter, the "Lee reference") in view of a 1995 article entitled "Dynamic Detection and Classification of Computer Viruses Using General Behavior Patterns," authored by Le Charlier et al. (hereinafter, the "Le Charlier reference"). The Examiner rejected Claims 2-3, and 5-6 under 35 U.S.C. § 103(a) as being unpatentable over the Lee reference in view of the Le Charlier reference, and further in view of a 1986 book entitled, "Advanced MS-DOS," authored by Ray Duncan (hereinafter, the "Duncan reference").

The Examiner rejected Claims 4, 7, and 10 under 35 U.S.C. § 103(a) as being unpatentable over the Lee reference in view of the Le Charlier reference, the Duncan reference, and further in view of a 1994 dictionary entitled, "IBM Dictionary of Computing," authored by George McDaniel (hereinafter, the "McDaniel reference"). The Examiner rejected Claims 8-9 under 35 U.S.C. § 103(a) as being unpatentable over the Lee reference in view of the Le Charlier reference, the Duncan reference, and further in view of 1998 book entitled, "Systems Architecture," authored by Stephen B. Burd (hereinafter, the "Burd reference") and a 1993 book entitled, "Inside Windows NT," authored by Helen Custer (hereinafter, the "Custer reference").

The Applicant respectfully offers remarks to traverse these pending rejections. The Applicant will address each independent claim separately as the Applicant believes that each independent claim is separately patentable over the prior art of record.

Independent Claim 11

It is respectfully submitted that the Lee, Le Charlier, Duncan, McDaniel, Burd, and Custer references, individually or in view of each other, fail to describe, teach, or suggest the combination of: (1) evaluating a file format of the target program; (2) evaluating control fields within a header of a file containing the target program; (3) automatically configuring the virtual machine to execute the target program in one of three modes of operation based on the file format and the control fields within the header of the file, (4) a first mode of operation comprising a real mode, (5) a second mode of operation for executing target programs comprising a high level programming language, and a (6) third mode of operation comprising a protected mode for executing target programs comprising thirty-two bit code; (7) storing behavior flags representing behavior of the target program during execution of the target program by the virtual machine; (8) storing a sequence in which the behavior flags are set by the target program during execution of the target program by the virtual machine; (9) passing behavior flag data and sequence flag data from the virtual machine to the computer system for evaluation after execution of the target program by the virtual machine; and (10) terminating the virtual machine after execution of the target program, thereby (11) removing from the computer system a copy of the target program that was contained within the virtual machine, as recited in new independent Claim 11.

The Lee Reference

The Lee reference describes a Virus Instruction Code Emulation (VICE) Methodology. This VICE methodology has two parts: an emulator and an analyzer. The emulator emulates the execution of a target file in a virtual system environment, and sends events that it collects to the analyzer. The analyzer is responsible for collecting incoming events and using a knowledge base to decide whether the events exhibit virus behavior. See Lee reference, page 214, first column, section B. "Components of Vice."

The Lee reference further explains that if the sequence of received events satisfies a rule in the knowledge base, the analyzer explains why the target file is a virus and what type of virus it is. The analyzer then instructs the emulator to stop generating events. See Lee reference, page 214, first column, section B. "Components of Vice."

Application Serial No. 09/885,427

While the Examiner alleges that the Lee reference teaches a virtual machine, the Examiner admits that the Lee reference does not describe storing a sequence in which the behavior flags are set by the target program during execution of the target program by the virtual machine. The Lee reference also does not provide any teaching of evaluating a file format of the target program; evaluating control fields within a header of a file containing the target program; automatically configuring the virtual machine to execute the target program in one of three modes of operation based on the file format and the control fields within the header of the file, a first mode of operation comprising a real mode, a second mode of operation for executing target programs comprising a high level programming language, a third mode of operation comprising a protected mode for executing target programs comprising thirty-two bit code, and passing behavior flag data and sequence flag data from the virtual machine to the computer system for evaluation after execution of the target program by the virtual machine, as recited in new independent Claim 11.

The Le Charlier Reference

To make up for the behavior flag sequencer deficiency of the Lee reference, the Examiner relies upon the Le Charlier reference. The Le Charlier reference describes how an emulator is used to monitor system activity of a virtual PC, and how an expert system ASAX is used to analyze the stream of data that the emulator produced.

The Le Charlier reference also describes how general rules are used to generally detect real viruses reliably, and specific rules to extract details of their behavior. See the Le Charlier reference, page 1.

Sections 4.5 and 4.6 of the Le Charlier reference describe an audit trail of an emulator. The audit trail includes parameters popped from a stack with return values. See Figure 3 of the Le Charlier reference reproduced below and page 15.

```

<CS=3911 Type=0 Fn=30 arg() ret( AX=5)>
<CS=3911 Type=0 Fn=29 arg() ret( BX=128 ES=3911)>
<CS=3911 Type=0 Fn=64 arg( AL=81 CL=3 stri=<.COM> ret( AL=0 CF=0)>
<CS=3911 Type=0 Fn=61 arg( AL=0 stri=COMMAND.COM) ret( AL=0 CX=32 CF=0)>
<CS=3911 Type=0 Fn=61 arg( AL=1 stri=COMMAND.COM) ret( AL=0 CX=32 CF=0)>
<CS=3911 Type=0 Fn=45 arg( AL=2 CL=32 stri=COMMAND.COM) ret( AL=0 AX=5 CF=0)>
<CS=3911 Type=0 Fn=73 arg( BX=5) ret( CX=10241 DX=6206 CF=0)>
<CS=3911 Type=0 Fn=27 arg() ret( CX=5121 DX=8032)>
<CS=3911 Type=0 Fn=47 arg( BX=5 CX=3 DX=628 DS=3911) ret( AX=3 CF=0)>
<CS=3911 Type=0 Fn=60 arg( AL=2 BX=5 CX=0 DX=0) ret( AL=0 AX=50031 DX=0 CF=0)>
<CS=3911 Type=0 Fn=48 arg( BX=5 CX=648 DX=313 DS=3911) ret( AX=648 CF=0)>
<CS=3911 Type=0 Fn=50 arg( AL=0 BX=5 CX=0 DX=0) ret( AL=0 AX=0 DX=0 CF=0)>
<CS=3911 Type=0 Fn=49 arg( BX=5 CX=3 DX=631 DS=3911) ret( AX=3 CF=0)>
<CS=3911 Type=0 Fn=74 arg( BX=5 CX=10271 DX=6206) ret( CF=0)>
<CS=3911 Type=0 Fn=46 arg( BX=5) ret( CF=0)>
<CS=3911 Type=0 Fn=51 arg( AL=1 stri=COMMAND.COM) ret( AL=0 CX=32 CF=0)>
:

```

Figure 3: Excerpt from an audit trail for the Vienna virus

Figure 3 of the Le Charlier reference is an example of an audit trail. The audit trail of this figure is a human representation of a binary NADF file. The example illustrated is from the Vienna virus.

Similar to the Lee reference, the Le Chartier reference also fails to provide any teaching of evaluating a file format of the target program; evaluating control fields within a header of a file containing the target program; automatically configuring the virtual machine to execute the target program in one of three modes of operation based on the file format and the control fields within the header of the file, a first mode of operation comprising a real mode, a second mode of operation for executing target programs comprising a high level programming language, a third mode of operation comprising a protected mode for executing target programs comprising thirty-two bit code, and passing behavior flag data and sequence flag data from the virtual machine to the computer system for evaluation after execution of the target program by the virtual machine, as recited in new independent Claim 11.

Application Serial No. 09/885,427

The Duncan Reference

The Examiner admits that the Lee and Le Charlier references do not provide any teaching of a structure that stores interrupt vector addresses and pointing at interrupt service routines loaded into memory reserved by the virtual machine when the virtual machine is initialized. To make up for this interrupt vector deficiency, the Examiner relies upon the Duncan reference.

The Duncan reference is a text book that describes the Microsoft Disk Operating System (DOS). The Examiner directs the Applicant's attention to page 207 and 208 of this reference for teachings of interrupt service routines and interrupt vector addresses.

The Applicant respectfully submits that interrupt service routines and interrupt vector addresses are in themselves not new and un-obvious. However, the Applicant submits that it is the combination of elements in each of the claims that distinguish the Applicant's technology from that of the prior art.

Similar to the Lee reference and the Le Charlier reference, the Duncan reference does not provide any teaching of evaluating a file format of the target program; evaluating control fields within a header of a file containing the target program; automatically configuring the virtual machine to execute the target program in one of three modes of operation based on the file format and the control fields within the header of the file, a first mode of operation comprising a real mode, a second mode of operation for executing target programs comprising a high level programming language, a third mode of operation comprising a protected mode for executing target programs comprising thirty-two bit code, and passing behavior flag data and sequence flag data from the virtual machine to the computer system for evaluation after execution of the target program by the virtual machine, as recited in new independent Claim 11.

The McDaniel Reference

The Examiner admits that the Lee reference, the Le Charlier reference, and the Duncan reference do not provide any teaching of a software processor that executes computer code under analysis by starting at each entry point defined within an entry point table and by producing a sequence in which behavior flags are set or reset. To address this behavior flag reset deficiency of these references, the Examiner relies upon the McDaniel reference.

The McDaniel reference is a technical dictionary that does not describe any virtual machine for detecting malicious computer code in target programs. The Examiner directs the

Application Serial No. 09/885,427

Applicant's attention to page 576 of the McDaniel reference that provides a generic definition for the term "reset." The Applicant respectfully submits that the Examiner is identifying isolated teachings in the art without considering the patentability of the combination of all the claimed elements.

The Applicant reminds the Examiner that the M.P.E.P. is clear on the subject of considering the differences between the claims and the prior art. The Examiner must consider the claimed combination as a whole as taught by M.P.E.P. § 2141.02 that states the following:

"THE CLAIMED INVENTION AS A WHOLE MUST BE
CONSIDERED."

In determining the differences between the prior art and the claims, the question under 35 U.S.C. § 103 is not whether the differences themselves would have been obvious, but whether the claimed invention as a whole would have been obvious. Stratoflex, Inc. v. Aeroquip Corp., 713 F.2d 1530, 218 USPQ 871 (Fed. Cir. 1983)." [Emphasis Supplied.]

The Applicant respectfully submits that the Examiner is over looking the specific design of Applicant's virtual machine invention for detecting malicious code and the generic designs presented by the prior art references that do not mention or even suggest virtual machines for detecting malicious computer code. For example, the Duncan and McDaniel references are only technical dictionaries that discuss the operations of real computers and not virtual ones. To address the virtual characteristics and the design of the Applicant's claimed technology, the Examiner relies upon these general dictionary references in combination with other references. The Applicant submits that the Examiner is not considering the claimed combination of elements as a whole, but is instead, he is looking at only the differences between the claimed combination and what elements exist in the prior art.

The Applicant also submits that McDaniel reference also does not provide any teaching of evaluating a file format of the target program; evaluating control fields within a header of a file containing the target program; automatically configuring the virtual machine to execute the target program in one of three modes of operation based on the file format and the control fields within the header of the file, a first mode of operation comprising a real mode, a second mode of operation for executing target programs comprising a high level programming language, a third

mode of operation comprising a protected mode for executing target programs comprising thirty-two bit code, and passing behavior flag data and sequence flag data from the virtual machine to the computer system for evaluation after execution of the target program by the virtual machine, as recited in new independent Claim 11.

The Burd Reference

The Examiner admits that the Lee, Le Charlier, and Duncan references do not provide any teaching of a software processor that executes 32-bit or 64-bit code. The Examiner also admits that these references also do not provide any teaching of an operating system simulation shell that responds to application program interface calls. To make up for the 32-bit and 64-bit code deficiency of the Lee, Le Charlier, and Duncan references, the Examiner relies upon the Burd reference.

The Examiner explains that the Burd reference is directed to computer systems architecture, including functions of operating systems, processor technology, and architecture. The Examiner further explains that pages 171 and 172 of the Burd reference teach 32-bit and 64-bit processors.

Similar to the analysis of the Duncan and McDaniel references, the Applicant is not claiming that virtual processors are new and un-obvious, rather, the Applicants are claiming that virtual processors in combination with all of the claim elements directed to detecting malicious code are patentable over the prior art.

The Applicant also submits that the Burd reference, being a general technical dictionary, does not provide any teaching of evaluating a file format of the target program; evaluating control fields within a header of a file containing the target program; automatically configuring the virtual machine to execute the target program in one of three modes of operation based on the file format and the control fields within the header of the file, a first mode of operation comprising a real mode, a second mode of operation for executing target programs comprising a high level programming language, a third mode of operation comprising a protected mode for executing target programs comprising thirty-two bit code, and passing behavior flag data and sequence flag data from the virtual machine to the computer system for evaluation after execution of the target program by the virtual machine, as recited in new independent Claim 11.

The Custer Reference

The Examiner admits that the Lee, Le Charlier, and Duncan references do not provide any teaching of a software processor that executes 32-bit or 64-bit code. The Examiner also admits that these references also do not provide any teaching of an operating system simulation shell that responds to application program interface calls. To make up for the 32-bit code and operating system shell deficiencies of the Lee, Le Charlier, and Duncan references, the Examiner relies upon the Custer reference.

The Examiner explains that since the Custer reference on page 150, second paragraph, describes a MS-DOS operating system being simulated, then it is obvious that the operating simulation shell responds to application program interface calls. Again, similar to the other general technical dictionary references like the McDaniel, Duncan, and Burd references, the Custer reference does not describe the virtual machine context of the present invention for detecting malicious computer code.

Specifically, the Custer reference, being a general technical document, does not provide any teaching of evaluating a file format of the target program; evaluating control fields within a header of a file containing the target program; automatically configuring the virtual machine to execute the target program in one of three modes of operation based on the file format and the control fields within the header of the file, a first mode of operation comprising a real mode, a second mode of operation for executing target programs comprising a high level programming language, a third mode of operation comprising a protected mode for executing target programs comprising thirty-two bit code, and passing behavior flag data and sequence flag data from the virtual machine to the computer system for evaluation after execution of the target program by the virtual machine, as recited in new independent Claim 11.

Conclusion Regarding Independent Claim 11

In light of the differences between Claim 11 and the Lee, Le Charlier, Duncan, McDaniel, Burd, and Custer references references mentioned above, one of ordinary skill in the art recognizes that the combination proposed by the Examiner cannot anticipate or render obvious the recitations as set forth in new independent Claim 11. Accordingly, consideration and allowance of new independent Claim 11 are respectfully requested.

Independent Claim 20

It is respectfully submitted that the Lee, Le Charlier, Duncan, McDaniel, Burd, and Custer references, individually or in view of each other, fail to describe, teach, or suggest the combination of: (1) a processing unit; (2) a memory storage device; and (3) one or more program modules stored in said memory storage device for providing instructions to said processing unit; said processing unit executing said instructions of said one or more program modules, operable for (4) evaluating a file format of the target program; (5) evaluating control fields within a header of a file containing the target program; (6) automatically configuring a virtual machine to execute the target program (7) in one of three modes of operation based on the file format and the control fields within the header of the file, (8) a first mode of operation comprising a real mode, (9) a second mode of operation for executing target programs comprising a high level programming language, and (10) third mode of operation for executing target programs comprising thirty-two bit code; storing behavior flags representing behavior of the target program during execution of the target program by the virtual machine; (11) storing a sequence in which the behavior flags are set by the target program of the target program during execution of the target program by the virtual machine; (12) passing behavior flag data and sequence flag data from the virtual machine to the computer system after execution of the target program by the virtual machine; and (13) evaluating the behavior flag data and sequence flag data with the computer system, as set forth in new independent Claim 20.

Similar to the analysis of new independent Claim 11, the Lee, Le Charlier, Duncan, McDaniel, Burd, and Custer references do not provide any teaching of evaluating a file format of the target program; evaluating control fields within a header of a file containing the target program; automatically configuring a virtual machine to execute the target program in one of three modes of operation based on the file format and the control fields within the header of the file, a first mode of operation comprising a real mode, a second mode of operation for executing target programs comprising a high level programming language, and third mode of operation for executing target programs comprising thirty-two bit code; storing behavior flags representing behavior of the target program during execution of the target program by the virtual machine, as recited in new independent Claim 20.

In light of the differences between Claim 20 and the Lee, Le Charlier, Duncan, McDaniel, Burd, and Custer references references mentioned above, one of ordinary skill in the

art recognizes that the combination proposed by the Examiner cannot anticipate or render obvious the recitations as set forth in new independent Claim 20. Accordingly, consideration and allowance of new independent Claim 20 are respectfully requested.

Independent Claim 29

It is respectfully submitted that the Lee, Le Charlier, Duncan, McDaniel, Burd, and Custer references, individually or in view of each other, fail to describe, teach, or suggest the combination of: (1) automatically configuring a virtual machine to execute the target program in one of three modes of operation, (2) a first mode of operation comprising a real mode, (3) a second mode of operation for executing a target program comprising a high level programming language, and (4) third mode of operation comprising a protected mode for executing a target program comprising thirty-two bit code; (5) storing behavior flags representing behavior of the target program during execution of the target program by the virtual machine; (6) storing a sequence in which the behavior flags are set by the target program of the target program during execution of the target program by the virtual machine; (7) passing behavior flag data and sequence flag data from the virtual machine to a computer system after execution of the target program by the virtual machine; and (8) terminating the virtual machine after execution of the target program, (9) thereby removing from the computer system a copy of the target program that was contained within the virtual machine, as recited in new independent Claim 29.

Similar to the analysis of new independent Claim 11, the Lee, Le Charlier, Duncan, McDaniel, Burd, and Custer references do not provide any teaching of automatically configuring a virtual machine to execute the target program in one of three modes of operation, a first mode of operation comprising a real mode, a second mode of operation for executing a target program comprising a high level programming language, and third mode of operation comprising a protected mode for executing a target program comprising thirty-two bit code; passing behavior flag data and sequence flag data from the virtual machine to the computer system after execution of the target program by the virtual machine; and terminating the virtual machine after execution of the target program, thereby removing from the computer system a copy of the target program that was contained within the virtual machine, as recited in new independent Claim 29.

In light of the differences between Claim 29 and the Lee, Le Charlier, Duncan, McDaniel, Burd, and Custer references mentioned above, one of ordinary skill in the

Application Serial No. 09/885,427

art recognizes that the combination proposed by the Examiner cannot anticipate or render obvious the recitations as set forth in new independent Claim 29. Accordingly, consideration and allowance of new independent Claim 29 are respectfully requested.

Dependent Claims 12-19, 21-28, and 30

The Applicant respectfully submits that the above-identified dependent claims are allowable because the independent claims from which they depend are patentable over the cited references. The Applicant also respectfully submits that the recitations of these dependent claims are of patentable significance.

In view of the foregoing, the Applicant respectfully requests that the Examiner withdraw the pending rejections of dependent Claims 12-19, 21-28, and 30.

CONCLUSION

The foregoing is submitted as a full and complete response to the Office Action mailed on January 4, 2005. The Applicant and the undersigned thank Examiner Guill for consideration of these remarks. The Applicant has amended the claims and has submitted new Claims 11-30. The Applicant respectfully submits that the present application is in condition for allowance. Such action is hereby courteously solicited.

If the Examiner believes that there are any issues that can be resolved by a telephone conference, or that there are any formalities that can be corrected by an Examiner's amendment, please contact the undersigned in the Atlanta Metropolitan area (404) 572-2884.

Respectfully submitted,

Steven P. Wigmore
Reg. No. 40,447

King & Spalding LLP
191 Peachtree Street, N.E.
Atlanta, Georgia 30303-1763
telephone: (404) 572.4600
K&S File No. 05456-105039